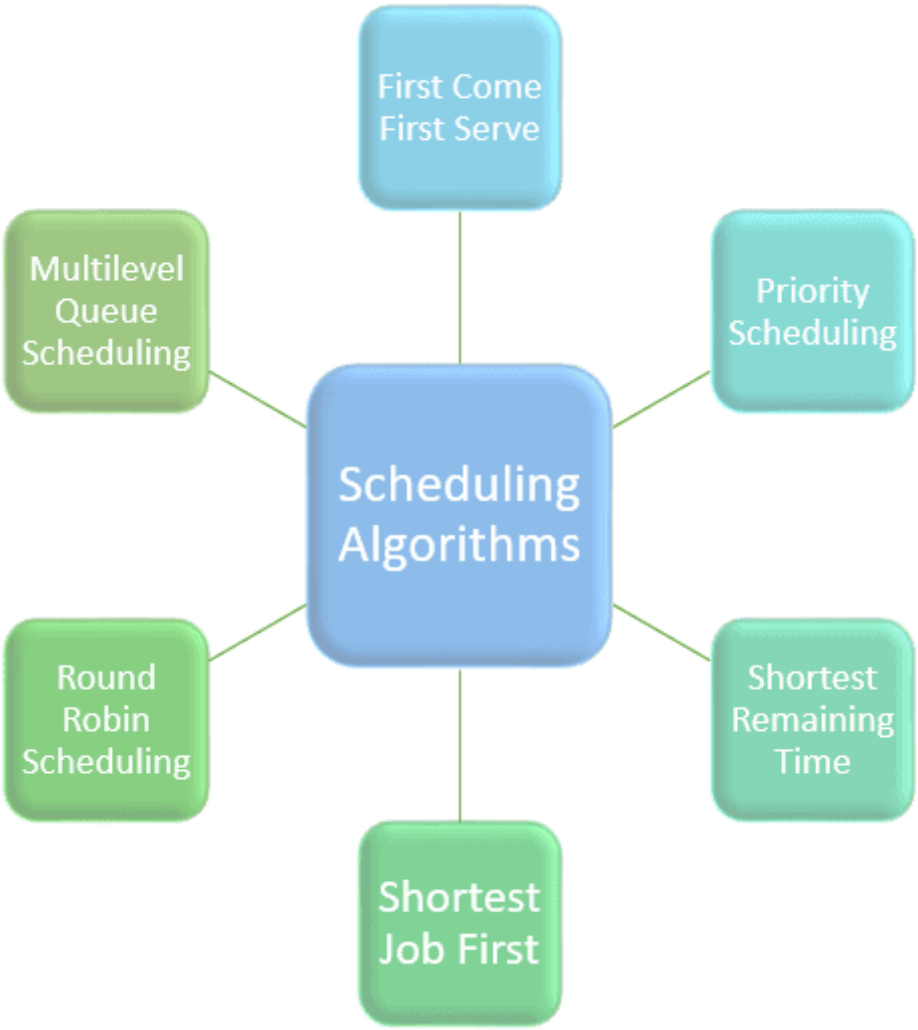
Λειτουργικά Συστήματα

Εξαμηνιαία Εργασία 2020-2021



Αναγνώστου Πανταζής 57497

Καρυπίδης Ευστάθιος 57556

Εισαγωγή

Σκοπός της εργασίας ήταν η προσομοίωση ενός λειτουργικού συστήματος μέσω της γλώσσας προγραμματισμού C. Ειδικότερα κληθήκαμε να υλοποιήσουμε και να συγκρίνουμε την πρακτική συμπεριφορά τεσσάρων από τους πίο δημοφιλείς αλγορίθμους χρονοδρομολόγησης/ χρονοπρογραμματισμού διεργασιών. Οι αλγόριθμοι αυτοί είναι ο FCFS(First Come First Serve), ο LRU(Least Recently Used), ο Round Robin και ο Premptive Priority. Αφού γίνει θεωρητική ανάλυση του προβλήματος και των επιμέρους αλγορίθμων, στην συνέχεια θα παρουσιαστεί ο κώδικας και η δομή του καθώς και τα παραδείγματα για τον έλεγχο του προγράμματος.

Μέρος Α: Θεωρητική Ανάλυση-Εξοικείωση

Πολύ συχνά στους σύγχρονους υπολογιστές συμβαίνει πολλές διεργασίες να ανταγωνίζονται για τον έλεγχο της CPU. Η κατάσταση αυτή εμφανίζεται όταν δύο η περισσότερες διεργασίες είναι έτοιμες να εκτελεστούν. Στην περίπτωση που είναι διαθέσιμη μία μόνο CPU τότε πρέπει να ληφθεί μία απόφαση για το ποιά θα εκτελεστή. Αυτό το αναλαμβάνει ένα τμήμα του λειτουργικού συστήματος το οποίο ονομάζεται χρονοπρογραμματιστής (scheduler) κάνοντας χρήση ενός αλγορίθμου που ονομάζεται αλγόριθμός χρονοπρογραμματισμού(scheduling algorithm). Συνοπτικά κάποιοι από τους λόγους για τις οποίες χρησιμοποιούνται αλγόριθμοι χρονοπρογραμματισμού είναι οι εξής:

- Βέλτιστη αξιοποίηση της CPU
- Δίκαιη κατανομή της CPU
- Μέγιστη διεκπεραιωτική ικανότητα(throughput)
- Ελάχιστο χρόνο διεκπεραίωσης, ελάχιστο χρόνο αναμονής, ελάχιστο χρόνο απόκρισης

Στην εκφώνηση δίνεται ο ακόλουθος πίνακας με ένα σύνολο διεργασιών καθώς και διάφορα χαρακτηριστικά τους. Ειδικότερα για κάθε διεργασία δίνεται (α) η μονάδα χρόνου άφιξής της, (β) ο χρόνος απασχόλησης της Κεντρικής Μονάδας Επεξεργασίας (σε μονάδες χρόνου) τον οποίο χρειάζεται η διαδικασία, για να ολοκληρωθεί και (γ) η προτεραιότητά της (όσο μεγαλύτερος είναι ο αντίστοιχος αριθμός για κάθε διαδικασία, τόσο μεγαλύτερη είναι η προτεραιότητά της)

Διαδικασία(Process)	**Χρόνος Άφιξης(Arrival Time)**	**Χρόνος Ολοκλήρωσης(Burst TIme)**	**Προτεραιότητα(Priority)**
P1	0	5	3
P2	1	6	5
P3	3	2	2
P4	9	4	1
P5	12	3	4

Με βάση το παραπάνω σύνολο διαδικασιών και για καθέναν από τους τέσσερις προαναφερθέντες αλγόριθμους χρονοδρομολόγησης, ζητείται να υλοποιηθούν ένας πίνακας της μορφής "Gantt chart" οποίος αποτελείται ουσιαστικά από δύο γραμμές, στην πρώτη από τις οποίες θα τυπώνονται οι διαδοχικές μονάδες χρόνου, ενώ στη δεύτερη θα τυπώνεται ο κωδικός αριθμός της διαδικασίας που χρησιμοποιεί την Κ.Μ.Ε. σε κάθε μονάδα χρόνου και ένα πίνακα μετρήσεων ο οποίος παρέχει (α) το χρόνο αναμονής, (β) το χρόνο διεκπεραίωσης, (γ) τον αριθμό εναλλαγών για κάθε διαδικασία, αφού ολοκληρωθεί η εκτέλεση όλων των διαδικασιών. Αρχικά ζητούνταν και και ο χρόνος απόκρισης αλλά σε διάλεξη αναφέρθηκε να μην υπολογιστεί.

Διαμοιρασμός Φόρτου Εργασίας

Αναγνώστου Πανταζής :

- Αλγόριθμοι First Come First Serve, Round Robin
- Δομή Process, Gantt
- Συγγραφή Report
- Ατομικά Παραδείγματα

Καρυπίδης Ευστάθιος :

- Αλγόριθμοι Priority based preemtive, Shortest Remaining Time First
- Δομή ReadyQueue, VirtualCPU
- Συγγραφή Report
- Ατομικά Παραδείγματα

Χρήσιμοι ορισμοί

Χρόνος Άφιξης(Arrival time) = Η στιγμή όπου η διεργασία μπαίνει σε ready state και είναι έτοιμη να εκτελεστεί.

Χρόνος/Διάρκεια Ολοκλήρωσης(Burst time) = Ο χρόνος που απαιτείται από τη CPU για την εκτέλεση.

Χρόνος Διεκπεραίωσης(Turn Around Time-**TAT**) = Ο χρόνος διεκπεραίωσης είναι μέσος(στατιστικά) χρόνος που μεσολαβεί μεταξύ της υποβολής(κατάσταση ready) και της ολοκλήρωσης μιας διεργασίας. Μπορεί να υπολογιστεί ως εξής. **TAT = Exit/Completion time - Arrival time= (Burst time + Waiting time)**

Χρόνος Αναμονής(Waiting time-**WT**) = Ο χρόνος τον οποίο μία διαδικασία βρίσκεται στην ready queue. Υπολογίζεται ως εξής: **WT = TAT - Burst time)**

Μη προεκτοπιστικοί αλγόριθμοι(non-preemtive) Σε κάθε διακοπή ρολογιού επιλέγουν μια διεργασία η οποία θα εκτελεστεί μέχρι να επιστρέψει εθελονικά τη CPU(η μέχρι να εμφανιστεί κάποια διακοπή κλπ).

Προεκτοπιστικοί αλγόριθμοι(preemtive) Σε καθε διακοπή ρολογιού επιλέγεται μια διεργασία η οποία θα εκτελεστεί και συνεπώς μια εργασία μπορεί να διακοπεί και ο επεξεργαστής να δοθεί σε μια νέα διεργασία.

FCFS(First Come First Serve)

Ο **First Come First Serve (FCFS)** είναι από τους απλούστερους αλγόριθμους και όπου υποδηλώνει το όνομά του, η πρώτη διεργασία που φθάνει (στην ready queue) "καταλαμβάνει " την CPU. Οι διεργασίες εξυπηρετούνται ανάλογα με το χρόνο άφιξης τους πράγμα που επιτυγχάνεται μέσω μιας ουράς FIFO. Παρά την ευκολία κατανόησης και υλοποιήσης ο FCFS δεν είναι ο ιδανικότερος και πιο αποδοτικός αλγόριθμος για διαμοιρασμό της CPU. Ειδικότερα μπορούν να εμφανιστούν εύκολα μεγάλοι χρόνοι αναμονής στην ready queue(να εμφανιστεί δηλαδή convoy effect) καθώς και το φαινόμενο όπου "σύντομες" διεργασίες να περιμένουν στην ουρά ώστε να τελειώσουν διεργασίες με μεγάλο χρόνο ολοκληρωσης. Τέλος, όπως είναι έυκολα κατανοητό είναι ένας μη προεκτοπιστικός αλγόριθμος.

Gantt	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20
	P1	P1	P1	P1	P1	P2	P2	P2	P2	P2	P2	P3	P3	P4	P4	P4	P4	P5	P5	P5

	P1	P2	P3	P4	P5
Χρόνος Διεκπεραίωσης	5-0=5	11-1=10	13-3=10	17-9=8	20-12=8
Χρόνος Αναμονής	5-5=0	10-6=4	10-2=8	8-4 = 4	8-3=5
Αριθμός εναλλαγών	0	0	0	0	0

Priority based preemtive (PP)

Ο **Priority Scheduling** είναι ένας αλγόριθμος χρονοπρογραμματισμού με βάση την προτεραιότητα, δηλαδή εκτελούνται πρώτα οι διεργασίες όπου έχουν μεγαλύτερη προτεραιότητα με αποτέλεσμα αυτές να μην χρειάζεται να περιμένουν πολύ στην ready queue. Ο αλγόριθμος είναι προεκτοπιστικός και είναι κατάλληλος όταν υπάρχει ανάγκη για καλύτερη "παροχή υπηρεσιών", με δεδομένο ότι η αναγαιότητα-προτεραιότητα των διαδικασιών είναι γνωστή. Ωστόσο, σε αυτήν την περίπτωση εάν οι διεργασίες υψηλής προτεραιότητας κατλαμβάνουν μεγάλο χρόνο στη CPU τότε οι διεργασίες χαμηλης προτεραιότητας συνεχώς αναβάλονται και συνεπώς εύκολα μπορεί να εμφανιστούν φαινόμενα όπως το starvation.

Gantt	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20
	P1	P2	P2	P2	P2	P2	P2	P1	P1	P1	P1	P3	P5	P5	P5	P3	P4	P4	P4	P4

	P1	P2	P3	P4	P5
Χρόνος Διεκπεραίωσης	11-0=11	7-1=6	16-3=13	20-9=11	15-12=3
Χρόνος Αναμονής	(7-1)=6	0	(11-3)+(15-12)=8+3=11	(16-9)=7	0
Αριθμός εναλλαγών	1	0	1	0	0

Shortest Remaining Time First (SRTF)

Ο **SRTF** αυτός είναι η "προεκτοπιστική" εκδοχή του αλγορίθμου **SJF(Sortest Job First)**. Όπως υποδηλώνει και το όνομά του οι διεργασίες με το μικρότερο χρόνο ολοκλήρωσης εκτελούνται πρώτες και μία διεργασία μπορεί να αντικατασταθεί εάν μία άλλη κατα την άφιξή της έχει μικρότερο χρόνο ολοκλήρωσης. Η απόδοση του συγκεκριμένου αλγορίθμου είναι πολύ καλή έως ιδανική(μικρός μέσος χρόνος αναμονής και διεκπεραίωσης). Ωστόσο, σε περισσότερες πρακτικές εφαρμογές είναι δύσκολο να είναι γνωστός απο πριν ο χρόνος ολοκλήρωσης. Ακόμη, μπορούν και εδώ να εμφανιστούν και εδώ φαινόμενα μεγάλης αναμονής και ειδικότερα για διεργασίες με μεγάλους χρόνους εκτέλεσης. Τέλος επιλέξαμε ότι σε περίπτωση ίσων χρόνων ολοκήρωσης να εκτελείται η διεργασία με μεγαλύτερο priority.

Gantt	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20
	P1	P1	P1	P1	P1	P3	P3	P2	P2	P2	P2	P2	P2	P5	P5	P5	P4	P4	P4	P4

	P1	P2	P3	P4	P5
Χρόνος Διεκπεραίωσης	5-0=5	13-1=12	7-3=4	20-9=11	16-12=4
Χρόνος Αναμονής	0	(7-1) = 6	(5-3)=2	(16-9)=7	(13-12)= 1
Αριθμός εναλλαγών	0	0	0	0	0

Round Robin (RR)

Ο αλγόριθμος **Round Robin** είναι απο τους πιο γνωστούς και παλιούς αλγορίθμους που μπορούν να υλοποιηθεί στα περισσότερα λειτουργικά συστήματα. Είναι προεκτοπιστικός αλγόριθμος και δίνει έμφαση στο διαμοιρασμό του χρόνου. Οι διεργασίες εκτελούνται με κυκλικό τρόπο για ένα συγκεκριμένο και ίσο χρονικό διάστημα η κάθε μία, το οποίο ονομάζεται κβάντο χρόνου. Εαν μια διεργασία ολοκληρωθεί σε αυτό το κβάνο τότε τερματίζεται αλλιώς επιστρέφει στην ready queue και περιμένει την επόμενη "επαναφορά/περιστροφή" να ολοκληρωθεί. Τα βασικά πλεονεκτήματα του συγκεκριμένου αλγορίθμου είναι ότι όλες οι διεργασίες διαμοιράζονται δίκαια την CPU, δεν εμφανίζονται φαινόμενα όπως το starvation η το convoy, δεν απαιτείται γνώση του χρόνου ολοκλήρωσης ή της προτεραιότητας και εμφανίζει καλό μέσο χρόνο αναμονής. Αντίθετα κάποια από τα μειονεκτήματα του αλγορίθμου είναι ότι δεν δίνεται κάποια προτεραιότητα σε "σημαντικές" διεργασίες, είναι δύσκολο να επιλεχθεί ένα "σωστό" κβάντου χρόνου καθώς επίσης η απόδοση του αλγορίθμου εξαρτάται πολύ απο την επιλογή του κβάντου χρόνου. Συνοπτικά, ένα μικρό κβάντο χρόνου μπορεί να οδηγήσει σε μεγαλύτερο overhead εναλλαγής σε ένα πραγματικό σύστημα ενώ ένα μεγάλο κβάντο χρόνου οδηγεί στη αύξηση του μέσου χρόνου αναμονής.Στο παράδειγμα που δώθηκε, ως κβάντο χρόνου ορίστηκαν οι 20 μονάδες(κατόπιν υπόδειξης) με αποτέλεσμα τα αποτελέσματα να είναι όμοια με τον FCFS.

Gantt	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20
	P1	P1	P1	P1	P1	P2	P2	P2	P2	P2	P2	P3	P3	P4	P4	P4	P4	P5	P5	P5

	P1	P2	P3	P4	P5
Χρόνος Διεκπεραίωσης	5-0=5	11-1=10	16-3=13	17-9=8	20-12=8
Χρόνος Αναμονής	5-5=0	10-6=4	10-2=8	8-4 = 4	8-3=5
Αριθμός εναλλαγών	0	0	0	0	0

Μέρος Β: Υλοποίηση

Ολοι οι αλγόριθμοι υλοποιήθηκαν σε ξεχωριστά header files ώστε να υπάρχει ευκολία στην κατανόηση τους. Πέρα των αλγορίθμων υπάρχει και ένα header file με κάποιες συναρτήσεις που χρησιμοποιούνται στο πρόγραμμα. Πιο συγκεκριμένα αυτή του sorting για τις διεργασίες που δίνονται από τον χρήστη. Ακόμη, στην λογική του κάθε αλγόριθμου δημιουργούνται οι δομές που περιγράψαμε παραπάνω και ακόμη γίνεται αρχικοποίηση τους στις μηδενικές τιμές που ορίσαμε οι ίδιοι.

Δομές:

Δομή Διεργασίας

Αρχικά, κατασκευάζουμε μία δομή της διεργασίας της οποίας θα επεξεργάζεται ο επεξεργαστής (**struct Process**). Αυτό το πρότυπο θα ακολουθούν όλες οι διεργασίες του συστήματος. Οι μεταβλητές που έχει η δομή της διεργασίας είναι οι παρακάτω:

- **name:** το οποίο είναι το πεδίο του ονόματος κάθε διεργασίας (μέχρι 3 χαρακτήρες).
- **arrive_time:** ο χρόνος που φτάνει κάθε διεργασία στον επεξεργαστή
- **burst_time:** ο χρόνος που θέλει μια διεργασία ώστε να ολοκληρωθεί
- **priority:** η προτεραιότητα σύμφωνα με την οποία πρέπει να διαχειριστεί η διεργασία

Σύμφωνα με αυτή τη δομή μπορούμε να φτιάξουμε όσες κενές διεργασίες θέλει ο χρήστης και στην συνέχεια να τις δώσει τις τιμές που ο ίδιος επιθυμεί.

Δομή ReadyQueue

Η δομή ReadyQueue είναι ουσιαστιά ένας πίνακας που αποτελείται από τις διεργασίες που έφτασαν στον επεξεργαστή αλλά δεν τρέχουν διότι ο επεξεργαστής είναι απασχολημένος. Ταξινομούνται μέσα στο πίνακα με βάση το ποιά διεργασία είναι αμέσως επόμενη να εκτελεστεί. Τα στοχεία που την αποτελούν είναι τα παρακάτω:

- **processes_waiting:** Πίνακας τύπου Process όπου αποθηκεύονται οι διεργασίες που περιμένουν

Δομή VirtualCpu

Παρομοίως η δομή του VirtualCpu χρησιμοποιείται για την προσομοίωση μιας εικονικής CPU. Συνολικά στο πρόγραμμα, ανεξαρτήτου αλγορίθμου, πρέπει να υπάρχουν ισάριθμες εικονικές CPU με τις διαφορετικές διεργασίες που έχουμε. Τέλος, για την προσομοίωσωη της εικονικής CPU επιλέχθηκε να αναπαρασταθεί ως ενα σύνολο από τις διαδοχικές χρονικές στιγμές που τρέχουν οι αλγόριθμοι καθώς και στο ποιά καταστάση βρίσκεται η εκάστοτε εικονική CPU εκείνη την στιγμή. Οι 4 καταστάσεις οι οποίες χρησιμοποιήθηκαν είναι οι παρακάτω:

- **N:** Η εικονική CPU δεν χρησιμοποιείτε καθόλου και η αντίστοιχη διεργασία δεν έχει φτάσει ακόμα
- **W:** Η διεργασία έφτασε στη CPU ωστόσο η CPU είναι απασχολημένη άρα και περιμένει
- **U:** Η διεργασία χρησιμοποιείτε από την CPU
- **S:** Η διεργασία σταμάτησε προσωρινά ώστε να εκτελεστεί μία άλλη

Δομή Gantt

Η δομή Gantt είναι υπεύθυνη για την προσομοίωση του Gantt chart καθώς και των χρόνων που ζητούνται αλλά και την παρουσιάση των δύο προηγούμενων δομών για κάθε χρονική στιγμή του CPU.

Αποτελείται από τις παρακάτω μεταβλητές:

- **algorithm_name:** κρατάει το όνομα του αλγορίθμου που χρησιμοποιήθηκε για την δημιουργία του Gantt
- **numberOfProcesses:** δηλώνει τον συνολικών διεργασιών που έχουμε
- **totalTime:** συνολικός χρόνος που χρειάζεται ώστε να εκτελεστούν όλες οι διεργασίες
- **states:** πίνακας τύπου Process ίσος με τον totalTime σε μέγεθος ώστε να υποδηλώνει την συνάρτηση που χρησιμοποείτε κάνε στιγμή από το CPU.
- **waiting_times:** πίνακας με τον χρόνο που θα χρειαστεί κάθε διεργασία μέχρι να εκτελεστεί πρώτη φορά απο το CPU
- **ta_times:** πίνακας με τους turnaround χρόνους κάθε διεργασίας
- **number_of_changes:** πίνακας με τον αριθμό των αλλαγών κάθε διεργασίας κατα την ολική εκτέλεση του προγράμματος
- **queues:** πίνακας τύπου ReadyQueue ώστε να έχουμε την κατάσταση της ReadyQueue για καθε χρονική στιγμή
- **cpus:** πίνακας τύπου VirtualCpu ώστε να έχουμε την κατάσταση της διεργασίας κάθε διεργασίας μέσω της εικονικής CPU

Αλγόριθμοι

Για κάθε αλγόριθμο προσθέσαμε στο τέλος και τον υπολογισμό του μέσου χρόνου αναμονής και μέσου χρόνου διεκπεραίωσης.

FCFS

Αφού κάνουμε ταξινόμηση των διεργασιών με αύξοντα χρόνο άφιξης υλοποιούμε τον αλγόριθμο. Σύμφωνα με αυτόν, η K.M.E. παίρνει μία μία τις διεργασίες και τις εκτελεί για τον αντιστοίχο χρόνο ολοκλήρωσης τους. Κατά την διάρκεια της εκτέλεσης κάθε διεργασίας συμπληρώνεται και το χρονοδιάγραμμα του Gantt σύμφωνα με το ποιά διεργασία εκτελείται κάθε στιγμή. Τέλος, αφού ολοκληθούν οι διεργασίες υπολογίζονται όλοι οι χρόνοι που αναφέρθηκαν και παραπάνω. Παράλληλα με την διαδικασία εκτέλεσης μιας διεργασίας γίνεται έλεγχος για την άφιξη των υπόλοιπων ώστε να συμπληρώνονται το ReadyQueue (διεργασίες που περιμένουν) καθώς και τα virtualCPUs κάθε διεργασίας. Στην κορυφή του ReadyQueue βρίσκεται η διεργασία που θα εκτελεστεί ως επόμενη ενώ το virtualCPU κάθε διεργασιάς αναπαριστά σε ποια κατάσταση βρίσκεται η ίδια για κάθε χρονική στιγμή. Να σημειωθεί πως σε αυτό τον αλγόριθμο μία διεργασία δεν μπορεί ποτέ να μπεί σε κατάσταση sleeping (γράμμα S στα παραδείγματα) αφού αν επιλεχθεί να εκτελεστεί απο την K.M.E. δεν ξαναβγαίνει μέχρι την ολοκλήρωση της.

Preemptive Priority

Για τον αλγόριθμο αυτό τρέχει ενα επαναλήπτικο κομμάτι όπου κάθε επανάληψη αναπαριστά την κάθε χρονική στιγμή της K.M.E. Κατά την διάρκεια της κάθε επανάληψης ο αλγόριθμος ελέγχει αρχικά αν έχει φτάσει μια νέα διεργασία και αν έχει μεγαλύτερη προτεραιότητα από αυτή της τωρινής (έχει οριστεί και μία "μηδενική" προτεραιότητα σε περίπτωση που δεν υπάρχει άλλη διεργασία στην K.M.E.). Αν ισχύουν τα προηγούμενα τότε επιλέγεται να εκτελεστεί αυτή η διεργασία για αυτή τη χρονική στιγμή, μειώνεται ο χρόνος ολοκλήρωσης της κατά ένα και στη συνέχεια μεταβαίνουμε στην επόμενη χρονική στιγμή. Όταν πλέον ο χρόνος όλοκλήρωσης όλων των διαδικασιών έχει φτάσει στο μηδέν τότε σημαίνει ότι ο αλγόριθμος έχει φτάσει στο τέλος του. Όπως και στον προηγούμενο αλγόριθμο άλλα και στους υπόλοιπους που θα παρουσιαστούν παρακάτω, παράλληλα με την εύρεση της τωρινής διεργασίας βρίσκουμε και ποιές διεργασίες έφτασαν αλλά δεν μπορούν να εξυπηρετηθούν αυτή τη χρονική στιγμή. Αυτές οι διεργασίες τοποθετούνται στη συνέχεια στην ReadyQueue πάλι με γνώμονα ποια θα εκτελεστεί στη συνέχεια. Ακόμη, συμπληρώνονται και τα virtualCPUs της κάθε διεργάσιας ανάλογα την κατάσταση στην οποία βρίσκεται η ίδια. Σε αυτό τον αλγόριθμο εν αντίθεση του προηγουμένου μία διεργασία μπορεί να βρίσκεται σε καταστάση sleeping αφού μπορεί να μπεί στην K.M.E και στη συνέχεια μία άλλη διεργασία να πάρει την θέση της.

RoundRobin

Όπως εξηγήθηκε και παραπάνω ο αλγόριθμος της RoundRobin χρησιμοποεί ένα κβάντο χρόνου ώστε να είναι διαμοιρασμένος σωστά ο χρόνος για κάθε διεργασία. Αυτό που κάνουμε στον αλγόριθμο είναι να εισάγουμε πάλι μια επαναληπτική διαδικασία η οποία ξεκινάει από την διεργασία που έφτασε πρώτη. Μετά ανάλογα τον υπολοιπόμενο χρόνο ολοκλήρωσης της μπαίνει σε μία απο τις δύο υλοποιήσεις. Αν ο υπολοιπόμενος χρόνος της είναι μικρότερος του κβάντου που δίνεται τότε η διεργασία υλοποιείται για τις υπόλοιπες μονάδες χρόνου της και στην συνέχεια μεταβαίνουμε στην επόμενη. Αν απο την άλλη είναι μεγαλύτερος του κβάντου τότε η διεργασία υλοποιείται μόνο για το κβάντο χρόνου και έπειτα μεταβαίνουμε στην επόμενη. Και στις 2 περιπτώσεις μείωνουμε τον χρόνο ολοκλήρωσης κατά πόσες φορές χρησιμοποιήθηκε η διεργασία ενώ λογικά προκύπτει πώς αν επιλεχθεί η 1η περίπτωση τότε και θα τελειώσει η συγκεκριμένη διεργασία. Η επιλογή της επόμενης διεργασίας που θα υλοποιηθεί γίνεται όπως πάντα με την κορυφή της ReadyQueue η οποία αναλογά και την περίπτωση που είμαστε συμπληρώνεται ανάλογα. Και σε αυτό τον αλγόριθμο έχουμε συμπλήρωση των virtualCPU ανάλογα την καταστάση στην οποία βρίσκεται κάθε διεργασία κάθε χρονική στιγμή ενώ οι χρόνοι(waiting, turnaround κλπ κλπ) υπολογίζονται με το που η αντίστοιχη διεργασία ολοκληρωθεί. Τέλος, αφού ολοκληρωθούν όλες οι διεργασίες κάνουμε έξοδο από τον αλγόριθμο μας επιστρέφοντας το διάγραμμα Gantt πλήρως συμπληρωμένο.

Σημέιωση: Κατά την συμπλήρωση του ReadyQueue κατά την δίαρκεια μιας αλλαγής, σε περίπτωση που φτάσει μια νέα διεργασία και έχουμε και την διεργασία που υλοποιούταν πιο πριν στην κορυφή του ReadyQueue μπαίνει η νέα ώς πιο δίκαιο σύστημα.

Shortest Remaining Time First (SRTF)

Όπως και στον Preemtive Priority τρέχει ένα επαναληπτικό κομμάτι όπου κάθε επανάληψη αναπαριστά την κάθε χρονική στιγμή της Κ.Μ.Ε. Κατα την διάρκεια των επαναλήψεων ελέγχει αν αρχικά φτάσει μία νέα διεργασία η οποία να έχει μικρότερο χρόνο ολοκλήρωσης. Σε περίπτωση ισότητας χρόνων ολοκλήρωσης επιλέγεται η διεργασία με την μεγαλύτερη προτεραιότητα. Με βάση αυτά επιλέγεται κάθε φορά η κατάλληλη διεργασία, μειώνεται ο χρόνος ολοκλήρωσης της κατά ένα και μεταβαίνουμε στην επόμενη χρονική στιμή. Όταν όλες οι εργασίες ολοκληρωθούν(ελέγχονται αν οι χρόνοι ολοκλήρωσης γίνουν 0) τότε ο αλγόριθμος έχει τελειώσει. Αντιστοιχα με τους άλλους αλγορίθμους βρίσκονται τόσο οι διεργασίες οι οποίες έφτασαν αλλα δεν μπόρεσαν να εξυπηρετηθούν η εξυπηρετούνται και διακόπηκαν με αποτέλεσμα να μπούν στο ready queue. Ετσι, συμπληρώνονται και τα VirtualCPUS της κάθε διεργασίας ανάλογα με την κατάσσταση που μπορεί να μπέι στην Κ.Μ.Ε. Και εδώ οι διεργασίες μπορούν να μπουν και σε κατάσταση sleeping σε περίπτωση που καταφθάσει νέα διεργασία και να πάρει την θέση της παλιάς.

Screenshot Λειτουργίας - Παράδειγμα εκφώνησης

Αρχικά δίνουμε ως είσοδο τα δεδομένα:

```
Enter number of processes:5
Enter name of process:p1
Enter arrive,burst time and priority of p1: 0 5 3
Enter name of process:p2
Enter arrive,burst time and priority of p2: 1 6 5
Enter name of process:p3
Enter arrive,burst time and priority of p3: 3 2 2
Enter name of process:p4
Enter arrive,burst time and priority of p4: 9 4 1
Enter name of process:p5
Enter arrive,burst time and priority of p5: 12 3 4
Enter time quantum:20
```

Τα αποτελέσματα είναι τα εξής:

```
Time:0    1    2    3    4    5    6    7    8    9    10   11   12   13   14   15   16   17   18   19   20
FCFS:| p1 | p1 | p1 | p1 | p1 | p2 | p2 | p2 | p2 | p2 | p2 | p3 | p3 | p4 | p4 | p4 | p4 | p5 | p5 | p5 |

Ready Queue:
| ::  |  p2  |  p2  |  p2  |  p2  |  p3  |  p3  |  p3  |  p3  |  p3  |  p3  |  p4  |  p4  |  p5  |  p5  |  p5  |  p5  |  ::  |  ::  |  ::  |
| ::  |  ::  |  ::  |  p3  |  p3  |  ::  |  ::  |  ::  |  ::  |  ::  |  ::  |  p4  |  p4  |  ::  |  p5  |  ::  |  ::  |  ::  |  ::  |  ::  |  ::  |  ::  |
| ::  |  ::  |  ::  |  ::  |  ::  |  ::  |  ::  |  ::  |  ::  |  ::  |  ::  |  ::  |  ::  |  ::  |  ::  |  ::  |  ::  |  ::  |  ::  |  ::  |  ::  |  ::  |
| ::  |  ::  |  ::  |  ::  |  ::  |  ::  |  ::  |  ::  |  ::  |  ::  |  ::  |  ::  |  ::  |  ::  |  ::  |  ::  |  ::  |  ::  |  ::  |  ::  |  ::  |  ::  |

Virtual CPUs:
Virtual CPU p1: |  U  |  U  |  U  |  U  |  U  |  N  |  N  |  N  |  N  |  N  |  N  |  N  |  N  |  N  |  N  |  N  |  N  |  N  |  N  |  N  |
Virtual CPU p2: |  N  |  W  |  W  |  W  |  W  |  U  |  U  |  U  |  U  |  U  |  U  |  N  |  N  |  N  |  N  |  N  |  N  |  N  |  N  |  N  |
Virtual CPU p3: |  N  |  N  |  N  |  W  |  W  |  W  |  W  |  W  |  W  |  W  |  W  |  U  |  U  |  N  |  N  |  N  |  N  |  N  |  N  |  N  |
Virtual CPU p4: |  N  |  N  |  N  |  N  |  N  |  N  |  N  |  N  |  N  |  N  |  W  |  W  |  W  |  W  |  U  |  U  |  U  |  U  |  N  |  N  |  N  |
Virtual CPU p5: |  N  |  N  |  N  |  N  |  N  |  N  |  N  |  N  |  N  |  N  |  N  |  N  |  N  |  W  |  W  |  W  |  W  |  W  |  U  |  U  |  U  |


| Time |  p1  |  p2  |  p3  |  p4  |  p5  |
|  WT  |  0   |  4   |  8   |  4   |  5   |
|  TA  |  5   |  10  |  10  |  8   |  8   |
|  NC  |  0   |  0   |  0   |  0   |  0   |

Average Waiting Time: 4.20
Average Turnaround Time: 8.20

Time:0    1    2    3    4    5    6    7    8    9    10   11   12   13   14   15   16   17   18   19   20
preE:| p1 | p2 | p2 | p2 | p2 | p2 | p2 | p1 | p1 | p1 | p1 | p3 | p5 | p5 | p5 | p3 | p4 | p4 | p4 | p4 |

Ready Queue:
| ::  |  p1  |  p1  |  p1  |  p1  |  p1  |  p1  |  p3  |  p3  |  p3  |  p3  |  p4  |  p3  |  p3  |  p3  |  p4  |  ::  |  ::  |  ::  |  ::  |
| ::  |  ::  |  ::  |  p3  |  p3  |  p3  |  p3  |  ::  |  ::  |  p4  |  p4  |  ::  |  p4  |  p4  |  p4  |  ::  |  ::  |  ::  |  ::  |  ::  |  ::  |
| ::  |  ::  |  ::  |  ::  |  ::  |  ::  |  ::  |  ::  |  ::  |  ::  |  ::  |  ::  |  ::  |  ::  |  ::  |  ::  |  ::  |  ::  |  ::  |  ::  |  ::  |
| ::  |  ::  |  ::  |  ::  |  ::  |  ::  |  ::  |  ::  |  ::  |  ::  |  ::  |  ::  |  ::  |  ::  |  ::  |  ::  |  ::  |  ::  |  ::  |  ::  |  ::  |

Virtual CPUs:
Virtual CPU p1: |  U  |  S  |  S  |  S  |  S  |  S  |  S  |  U  |  U  |  U  |  U  |  N  |  N  |  N  |  N  |  N  |  N  |  N  |  N  |  N  |
Virtual CPU p2: |  N  |  U  |  U  |  U  |  U  |  U  |  U  |  N  |  N  |  N  |  N  |  N  |  N  |  N  |  N  |  N  |  N  |  N  |  N  |  N  |
Virtual CPU p3: |  N  |  N  |  N  |  W  |  W  |  W  |  W  |  W  |  W  |  W  |  W  |  U  |  S  |  S  |  S  |  U  |  N  |  N  |  N  |  N  |
Virtual CPU p4: |  N  |  N  |  N  |  N  |  N  |  N  |  N  |  N  |  N  |  N  |  W  |  W  |  W  |  W  |  W  |  W  |  U  |  U  |  U  |  U  |
Virtual CPU p5: |  N  |  N  |  N  |  N  |  N  |  N  |  N  |  N  |  N  |  N  |  N  |  N  |  N  |  U  |  U  |  U  |  N  |  N  |  N  |  N  |


| Time |  p1  |  p2  |  p3  |  p4  |  p5  |
|  WT  |  6   |  0   |  11  |  7   |  0   |
|  TA  |  11  |  6   |  13  |  11  |  3   |
|  NC  |  1   |  0   |  1   |  0   |  0   |

Average Waiting Time: 4.80
Average Turnaround Time: 8.80
```

```
Time:0   1   2   3   4   5   6   7   8   9   10  11  12  13  14  15  16  17  18  19  20
RoRo:| p1 | p1 | p1 | p1 | p1 | p2 | p2 | p2 | p2 | p2 | p2 | p3 | p3 | p4 | p4 | p4 | p4 | p5 | p5 | p5 |

Ready Queue:
| ::  | p2  | p2  | p2  | p2  | p3  | p3  | p3  | p3  | p3  | p3  | p4  | p4  | p5  | p5  | p5  | p5  | ::  | ::  | ::  |
| ::  | ::  | ::  | p3  | p3  | ::  | ::  | ::  | ::  | ::  | p4  | p4  | ::  | p5  | ::  | ::  | ::  | ::  | ::  | ::  | ::  |
| ::  | ::  | ::  | ::  | ::  | ::  | ::  | ::  | ::  | ::  | ::  | ::  | ::  | ::  | ::  | ::  | ::  | ::  | ::  | ::  | ::  |
| ::  | ::  | ::  | ::  | ::  | ::  | ::  | ::  | ::  | ::  | ::  | ::  | ::  | ::  | ::  | ::  | ::  | ::  | ::  | ::  | ::  |

Virtual CPUs:
Virtual CPU p1: | U | U | U | U | U | N | N | N | N | N | N | N | N | N | N | N | N | N | N | N |
Virtual CPU p2: | N | W | W | W | W | U | U | U | U | U | U | N | N | N | N | N | N | N | N | N |
Virtual CPU p3: | N | N | N | W | W | W | W | W | W | W | W | U | U | N | N | N | N | N | N | N |
Virtual CPU p4: | N | N | N | N | N | N | N | N | N | W | W | W | W | U | U | U | U | N | N | N |
Virtual CPU p5: | N | N | N | N | N | N | N | N | N | N | N | N | W | W | W | W | W | U | U | U |


| Time |  p1  |  p2  |  p3  |  p4  |  p5  |
|  WT  |  0   |  4   |  8   |  4   |  5   |
|  TA  |  5   |  10  |  10  |  8   |  8   |
|  NC  |  0   |  0   |  0   |  0   |  0   |

Average Waiting Time: 4.20
Average Turnaround Time: 8.20

Time:0   1   2   3   4   5   6   7   8   9   10  11  12  13  14  15  16  17  18  19  20
SRTF:| p1 | p1 | p1 | p1 | p3 | p3 | p2 | p2 | p2 | p2 | p2 | p2 | p5 | p5 | p5 | p4 | p4 | p4 | p4 |

Ready Queue:
| ::  | p2  | p2  | p3  | p3  | p2  | p2  | ::  | ::  | p4  | p4  | p4  | p5  | p4  | p4  | p4  | ::  | ::  | ::  | ::  |
| ::  | ::  | ::  | p2  | p2  | ::  | ::  | ::  | ::  | ::  | ::  | ::  | p4  | ::  | ::  | ::  | ::  | ::  | ::  | ::  | ::  |
| ::  | ::  | ::  | ::  | ::  | ::  | ::  | ::  | ::  | ::  | ::  | ::  | ::  | ::  | ::  | ::  | ::  | ::  | ::  | ::  | ::  |
| ::  | ::  | ::  | ::  | ::  | ::  | ::  | ::  | ::  | ::  | ::  | ::  | ::  | ::  | ::  | ::  | ::  | ::  | ::  | ::  | ::  |

Virtual CPUs:
Virtual CPU p1: | U | U | U | U | U | N | N | N | N | N | N | N | N | N | N | N | N | N | N | N |
Virtual CPU p2: | N | W | W | W | W | W | W | U | U | U | U | U | U | N | N | N | N | N | N | N |
Virtual CPU p3: | N | N | N | W | W | U | U | N | N | N | N | N | N | N | N | N | N | N | N | N |
Virtual CPU p4: | N | N | N | N | N | N | N | N | N | W | W | W | W | W | W | W | U | U | U | U |
Virtual CPU p5: | N | N | N | N | N | N | N | N | N | N | N | N | N | W | U | U | U | N | N | N | N |


| Time |  p1  |  p2  |  p3  |  p4  |  p5  |
|  WT  |  0   |  6   |  2   |  7   |  1   |
|  TA  |  5   |  12  |  4   |  11  |  4   |
|  NC  |  0   |  0   |  0   |  0   |  0   |

Average Waiting Time: 3.20
Average Turnaround Time: 7.20
```

Προσωπικό Παράδειγμα

Οσον αφορά το ατομικό μου παράδειγμα είναι:

Διαδικασία(Process)	Χρόνος Άφιξης(Arrival Time)	Χρόνος Ολοκλήρωσης(Burst TIme)	Προτεραιότητα(Priority)
P1	0	8	4
P2	2	4	1
P3	3	2	5
P4	7	3	6
P5	13	1	3
P6	14	5	2

Ως κβάντο χρόνου για τον Round Robin θεωρούμε 2 μονάδες χρόνου.

Τα αποτελέσματα για του διαγράμματος Gantt για όλους τους αλγορίθμους είναι:

Gannt	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20	21	22	23
FCFS	P1	P1	P1	P1	P1	P1	P1	P1	P2	P2	P2	P2	P3	P3	P4	P4	P4	P5	P6	P6	P6	P6	P6
preE	P1	P1	P1	P3	P3	P1	P1	P4	P4	P4	P1	P1	P1	P5	P6	P6	P6	P6	P6	P2	P2	P2	P2
RoRo	P1	P1	P1	P1	P2	P2	P3	P3	P1	P1	P2	P2	P4	P4	P1	P1	P5	P6	P6	P4	P6	P6	P6
SRTF	P1	P1	P2	P3	P3	P2	P2	P2	P4	P4	P4	P1	P1	P5	P1	P1	P1	P1	P6	P6	P6	P6	P6

Για τους χρόνους αναμονής και διεκπεραίωσης και αριθμούς εναλλαγών έχουμε

First Come First Serve	P1	P2	P3	P4	P5	P6
Χρόνος Διεκπεραίωσης	8-0=8	12-2=10	14-3=11	17-7=10	18-13=5	23-14=9
Χρόνος Αναμονής	0	(8-2)=6	(12-3)=9	(14-7)=7	(17-13)=4	(18-14)=4
Αριθμός εναλλαγών	0	0	0	0	0	0

Παρατηρούμε ότι οι χρόνοι διεκπεραίσης και αναμονής είναι υψηλοί ειδικά μετά την πρώτη διεργασία πράγμα που οφείλεται στο γεγονός οτι η πρώτη διεργασία είχε μεγάλο χρόνο ολοκλήρωσης και στο μεταξύ ήρθαν και άλλες διεργασίες και να περιμένουν(convoy effect). Ο αριθμός εναλλαγών όπως ειναι αναμενόμενο είναι 0 καθώς ο FCFS είναι μη προεκτοπιστικός.

Priority Preemtive	P1	P2	P3	P4	P5	P6
Χρόνος Διεκπεραίωσης	13-0=13	23-2=21	5-3=2	10-7=3	14-13=1	19-14=5
Χρόνος Αναμονής	(5-3)+(10-7)=5	(19-2)=17	0	0	0	0
Αριθμός εναλλαγών	2	0	0	0	0	0

Στον Priority-Based Preemtive παρατηρούμε πως οι χρόνοι αναμονής γενικά είναι μικροί πλην της διεργασίας 2. Αυτό συμβαίνει καθώς η 2η διεργασία έχει πολύ χαμηλό priority και συνεπώς καθώς φτάνουν διεργασίες παίρνουν συνεχώς τη σειρά της με αποτέλεσμα να περιμένει 17 μονάδες χρόνου ενώ αυτή απαιτεί 4 για την ολοκληρωσή της(οδηγείται δηλαδή σε starvation). Σε γενικές γραμμές ωστόσο τα αποτελέσματα του αλγορίθμου είναι πολύ καλά, αλλά για να υλοποιηθεί στην πραγματικότητα απαιτείται γνώση του priority των διεργασιών.

Round Robin	P1	P2	P3	P4	P5	p6
Χρόνος Διεκπεραίωσης	16-0=16	12-2=10	8-3=5	20-7=13	17-13=4	23-14=9
Χρόνος Αναμονής	(8-4)+(14-10)=8	(4-2)+(10-6)=6	(6-3)=3	(12-7)+(19-14)=10	(16-13)=3	(17-14)+(20-19)=4
Αριθμός εναλλαγών	2	1	0	1	0	1

Οπως παρατηρούμε τα αποτελέσματα του Round Robin όσον αφορά τους χρόνους διεκπεραίωσης και αναμονής είναι σχετικά μεγάλοι για διεργασίες που απαιτούν περισσότερο χρόνο από οτι το κβάντο χρόνου. Ακόμη για τις διεργασίες P3 και P5 παρατηρούμε ότι δεν εμφανίζεται καμία εναλλαγή πράγμα που συμβαίνει καθώς ο χρόνος ολοκλήρωσης τους ειναι μικρότερος από το κβάντο χρόνου. Ωστόσο οπως βλέπουμε δεν δημιουργούνται φαινόμενα όπως το convoy και υπάρχει σχετικά δίκαιος διαμοιρασμός της CPU.

Shortest Remaining Time First	P1	P2	P3	P4	P5	p6
Χρόνος Διεκπεραίωσης	18-0=18	8-2=6	5-3=2	11-7=4	14-13=1	23-14=9
Χρόνος Αναμονής	(11-2)+(14-13)=10	(6-4)=2	0	(8-7)=1	0	(18-14)=4
Αριθμός εναλλαγών	2	1	0	0	0	0

Στην περίπτωση του SRTF παρατηρούμε εξαιρετικά χαμηλό μέσο χρόνο αναμονής καθώς και αρκετά μικρό χρόνο διεκπεραίωσης. Οι μόνες διεργασίες που παρουσιάζουν σχετικά μεγάλους χρόνους αναμονής είναι η P1 και P6 οι οποίες όμως έχουν και μεγάλο χρόνο ολοκλήρωσης. Στην πραγματικότητα, ωστόσο είναι δύσκολο να υλοποιηθεί ένας τέτοιος αλγόριθμος καθώς απαιτεί την γνώση του χρόνου ολοκλήρωσης.

Στην συνέχεια παραθέτω τα screenshot από τις οθόνες λειτουργίας του προγράματος

```
Enter number of processes:6
Enter name of process:P1
Enter arrive,burst time and priority of P1: 0 8 4
Enter name of process:P2
Enter arrive,burst time and priority of P2: 2 4 1
Enter name of process:P3
Enter arrive,burst time and priority of P3: 3 2 5
Enter name of process:P4
Enter arrive,burst time and priority of P4: 7 3 6
Enter name of process:P5
Enter arrive,burst time and priority of P5: 13 1 3
Enter name of process:P6
Enter arrive,burst time and priority of P6: 14 5 2
Enter time quantum:2
```

```
Time:0    1    2    3    4    5    6    7    8    9    10   11   12   13   14   15   16   17   18   19   20   21   22   23
FCFS:| P1 | P1 | P1 | P1 | P1 | P1 | P1 | P1 | P2 | P2 | P2 | P2 | P3 | P3 | P4 | P4 | P4 | P5 | P6 | P6 | P6 | P6 | P6 |

Ready Queue:
| :: | :: | P2 | P2 | P2 | P2 | P2 | P2 | P3 | P3 | P3 | P3 | P4 | P4 | P5 | P5 | P5 | P6 | :: | :: | :: | :: | :: |
| :: | :: | :: | P3 | P3 | P3 | P3 | P3 | P4 | P4 | P4 | P4 | :: | P5 | P6 | P6 | P6 | :: | :: | :: | :: | :: | :: |
| :: | :: | :: | :: | :: | :: | :: | P4 | :: | :: | :: | :: | :: | :: | :: | :: | :: | :: | :: | :: | :: | :: | :: |
| :: | :: | :: | :: | :: | :: | :: | :: | :: | :: | :: | :: | :: | :: | :: | :: | :: | :: | :: | :: | :: | :: | :: |
| :: | :: | :: | :: | :: | :: | :: | :: | :: | :: | :: | :: | :: | :: | :: | :: | :: | :: | :: | :: | :: | :: | :: |

Virtual CPUs:
Virtual CPU P1: | U | U | U | U | U | U | U | U | N | N | N | N | N | N | N | N | N | N | N | N | N | N | N |
Virtual CPU P2: | N | N | W | W | W | W | W | W | U | U | U | U | N | N | N | N | N | N | N | N | N | N | N |
Virtual CPU P3: | N | N | N | W | W | W | W | W | W | W | W | W | U | U | N | N | N | N | N | N | N | N | N |
Virtual CPU P4: | N | N | N | N | N | N | N | W | W | W | W | W | W | W | U | U | U | N | N | N | N | N | N |
Virtual CPU P5: | N | N | N | N | N | N | N | N | N | N | N | N | N | W | W | W | W | U | N | N | N | N | N |
Virtual CPU P6: | N | N | N | N | N | N | N | N | N | N | N | N | N | N | W | W | W | W | U | U | U | U | U |


| Time | P1 | P2 | P3 | P4 | P5 | P6 |
|  WT  | 0  | 6  | 9  | 7  | 4  | 4  |
|  TA  | 8  | 10 | 11 | 10 | 5  | 9  |
|  NC  | 0  | 0  | 0  | 0  | 0  | 0  |

Average Waiting Time: 5.00
Average Turnaround Time: 8.83

Time:0    1    2    3    4    5    6    7    8    9    10   11   12   13   14   15   16   17   18   19   20   21   22   23
preE:| P1 | P1 | P1 | P3 | P3 | P1 | P1 | P4 | P4 | P4 | P1 | P1 | P1 | P5 | P6 | P6 | P6 | P6 | P6 | P2 | P2 | P2 | P2 |

Ready Queue:
| :: | :: | P2 | P1 | P1 | P2 | P2 | P1 | P1 | P1 | P2 | P2 | P2 | P2 | P2 | P2 | P2 | P2 | P2 | :: | :: | :: | :: |
| :: | :: | :: | P2 | P2 | :: | :: | P2 | P2 | P2 | :: | :: | :: | :: | :: | :: | :: | :: | :: | :: | :: | :: | :: |
| :: | :: | :: | :: | :: | :: | :: | :: | :: | :: | :: | :: | :: | :: | :: | :: | :: | :: | :: | :: | :: | :: | :: |
| :: | :: | :: | :: | :: | :: | :: | :: | :: | :: | :: | :: | :: | :: | :: | :: | :: | :: | :: | :: | :: | :: | :: |
| :: | :: | :: | :: | :: | :: | :: | :: | :: | :: | :: | :: | :: | :: | :: | :: | :: | :: | :: | :: | :: | :: | :: |

Virtual CPUs:
Virtual CPU P1: | U | U | U | S | S | U | U | S | S | S | U | U | U | N | N | N | N | N | N | N | N | N | N |
Virtual CPU P2: | N | N | W | W | W | W | W | W | W | W | W | W | W | W | W | W | W | W | W | U | U | U | U |
Virtual CPU P3: | N | N | N | U | U | N | N | N | N | N | N | N | N | N | N | N | N | N | N | N | N | N | N |
Virtual CPU P4: | N | N | N | N | N | N | N | U | U | U | N | N | N | N | N | N | N | N | N | N | N | N | N |
Virtual CPU P5: | N | N | N | N | N | N | N | N | N | N | N | N | N | U | N | N | N | N | N | N | N | N | N |
Virtual CPU P6: | N | N | N | N | N | N | N | N | N | N | N | N | N | N | U | U | U | U | U | N | N | N | N |


| Time | P1 | P2 | P3 | P4 | P5 | P6 |
|  WT  | 5  | 17 | 0  | 0  | 0  | 0  |
|  TA  | 13 | 21 | 2  | 3  | 1  | 5  |
|  NC  | 2  | 0  | 0  | 0  | 0  | 0  |

Average Waiting Time: 3.67
Average Turnaround Time: 7.50

Time:0    1    2    3    4    5    6    7    8    9    10   11   12   13   14   15   16   17   18   19   20   21   22   23
RoRo:| P1 | P1 | P1 | P1 | P2 | P2 | P3 | P3 | P1 | P1 | P2 | P2 | P4 | P4 | P1 | P1 | P5 | P6 | P6 | P4 | P6 | P6 | P6 |

Ready Queue:
| :: | :: | P2 | P2 | P3 | P3 | P1 | P1 | P2 | P2 | P4 | P4 | P1 | P1 | P5 | P5 | P6 | P4 | P4 | P6 | :: | :: | :: |
| :: | :: | :: | P3 | P1 | P1 | P2 | P2 | P4 | P4 | P1 | P1 | :: | P5 | P6 | P6 | P4 | :: | :: | :: | :: | :: | :: |
| :: | :: | :: | :: | :: | :: | :: | P4 | :: | :: | :: | :: | :: | :: | P4 | P4 | :: | :: | :: | :: | :: | :: | :: |
| :: | :: | :: | :: | :: | :: | :: | :: | :: | :: | :: | :: | :: | :: | :: | :: | :: | :: | :: | :: | :: | :: | :: |
| :: | :: | :: | :: | :: | :: | :: | :: | :: | :: | :: | :: | :: | :: | :: | :: | :: | :: | :: | :: | :: | :: | :: |

Virtual CPUs:
Virtual CPU P1: | U | U | U | U | S | S | S | S | U | U | S | S | S | S | U | U | N | N | N | N | N | N | N |
Virtual CPU P2: | N | N | W | W | U | U | S | S | S | S | U | U | N | N | N | N | N | N | N | N | N | N | N |
Virtual CPU P3: | N | N | N | W | W | W | U | U | N | N | N | N | N | N | N | N | N | N | N | N | N | N | N |
Virtual CPU P4: | N | N | N | N | N | N | N | W | W | W | W | W | U | U | S | S | S | S | S | U | N | N | N |
Virtual CPU P5: | N | N | N | N | N | N | N | N | N | N | N | N | N | W | W | W | U | N | N | N | N | N | N |
Virtual CPU P6: | N | N | N | N | N | N | N | N | N | N | N | N | N | N | W | W | W | U | U | S | U | U | U |


| Time | P1 | P2 | P3 | P4 | P5 | P6 |
|  WT  | 8  | 6  | 3  | 10 | 3  | 4  |
|  TA  | 16 | 10 | 5  | 13 | 4  | 9  |
|  NC  | 2  | 1  | 0  | 1  | 0  | 1  |

Average Waiting Time: 5.67
Average Turnaround Time: 9.50

Time:0    1    2    3    4    5    6    7    8    9    10   11   12   13   14   15   16   17   18   19   20   21   22   23
SRTF:| P1 | P1 | P2 | P3 | P3 | P2 | P2 | P2 | P4 | P4 | P4 | P1 | P1 | P5 | P1 | P1 | P1 | P1 | P6 | P6 | P6 | P6 | P6 |

Ready Queue:
| :: | :: | P1 | P2 | P2 | P1 | P1 | P4 | P1 | P1 | P1 | :: | :: | P1 | P6 | P6 | P6 | P6 | :: | :: | :: | :: | :: |
| :: | :: | :: | P1 | P1 | :: | :: | P1 | :: | :: | :: | :: | :: | :: | :: | :: | :: | :: | :: | :: | :: | :: | :: |
| :: | :: | :: | :: | :: | :: | :: | :: | :: | :: | :: | :: | :: | :: | :: | :: | :: | :: | :: | :: | :: | :: | :: |
| :: | :: | :: | :: | :: | :: | :: | :: | :: | :: | :: | :: | :: | :: | :: | :: | :: | :: | :: | :: | :: | :: | :: |
| :: | :: | :: | :: | :: | :: | :: | :: | :: | :: | :: | :: | :: | :: | :: | :: | :: | :: | :: | :: | :: | :: | :: |

Virtual CPUs:
Virtual CPU P1: | U | U | S | S | S | S | S | S | S | S | S | U | U | S | U | U | U | U | N | N | N | N | N |
Virtual CPU P2: | N | N | U | S | S | U | U | U | N | N | N | N | N | N | N | N | N | N | N | N | N | N | N |
Virtual CPU P3: | N | N | N | U | U | N | N | N | N | N | N | N | N | N | N | N | N | N | N | N | N | N | N |
Virtual CPU P4: | N | N | N | N | N | N | N | W | U | U | U | N | N | N | N | N | N | N | N | N | N | N | N |
Virtual CPU P5: | N | N | N | N | N | N | N | N | N | N | N | N | N | U | N | N | N | N | N | N | N | N | N |
Virtual CPU P6: | N | N | N | N | N | N | N | N | N | N | N | N | N | N | W | W | W | W | U | U | U | U | U |


| Time | P1 | P2 | P3 | P4 | P5 | P6 |
|  WT  | 10 | 2  | 0  | 1  | 0  | 4  |
|  TA  | 18 | 6  | 2  | 4  | 1  | 9  |
|  NC  | 2  | 1  | 0  | 0  | 0  | 0  |

Average Waiting Time: 2.83
Average Turnaround Time: 6.67
```

Πηγές

1. https://www.guru99.com/cpu-scheduling-algorithms.html#9
2. https://www.guru99.com/fcfs-scheduling.html
3. https://www.guru99.com/round-robin-scheduling-example.html
4. https://www.guru99.com/priority-scheduling-program.html
5. https://www.guru99.com/shortest-job-first-sjf-scheduling.html
6. https://www.javatpoint.com/os-scheduling-algorithms
7. https://www.tutorialspoint.com/operating_system/os_process_scheduling_algorithms.htm
8. https://afteracademy.com/blog/what-is-burst-arrival-exit-response-waiting-turnaround-time-and-throughput
9. Tanenbaum, Andrew S_Bos, Herbert - Modern operating systems - Pearson Education (Us)_ Langara College (2016)
10. Διαφάνιες Μαθήματος: Διάλεξη 9-10 Χρονοπρογραμματισμός opsys5-scheduling